

Throwing Ethernet Frames

You've come to the right place if you're looking for Ethernet technology that will enable you to remotely communicate with your embedded designs. This month Fred introduces you to the AirDrop-P/Frame Thrower module.

Say what you will about microcontrollers and Ethernet, but I get e-mails all the time telling me about how you guys and gals are using the "old man" in your modern microcontroller applications. Many of the e-mails are from industry professionals detailing large networks of small microcontroller-based Ethernet controllers collecting manufacturing data or helping to monitor production processes. The bulk of my incoming communiqués are from college students all over the world who are piecing together Ethernet engines and microcontrollers for their senior projects. As long as I continue to receive those types of messages, I'll keep one eye focused on new developments in Ethernet technology that you can apply to microcontrollers.

For the past couple of months, I've been reading about a new Ethernet controller with a radically innovative old-school interface. Instead of the typical address bus/data bus parallel interface, Microchip Technology's ENC28J60 Ethernet controller uses a SPI as the data pipe. The simplicity of a SPI hookup opens up a huge amount of Ethernet retrofit possibilities for mature microcontroller products that are I/O constrained.

I'm extremely familiar with the de facto standard Realtek RTL8019AS and its faster first cousin the ASIX AX88796L Ethernet controller. Both are built around the NE2000 register set. They also use a parallel address/data interface and employ a circular buffer scheme to move data between the twisted pair and the microcontroller. I

was pleasantly surprised to find that the ENC28J60, although much less complicated in terms of the interface, complements the RTL8019AS and AX88796L parts in the way it looks to the programmer. Let's take a closer look at the world's smallest Ethernet controller.

ENC28J60

A 28-pin package houses the ENC28J60 circuitry, whose assemblage of Ethernet subsystems collectively forms an IEEE 802.3-compatible Ethernet controller. In addition to the 10-Mbps SPI, the ENC28J60's 28 pins provide access to an integrated MAC and 10BaseT PHY. Like the RTL8019AS and AX88796L parts, the ENC28J60 uses a set of registers to configure its operational aspects. Unlike the RTL8019AS, Full Duplex and Half Duplex modes can be dialed in without having to include or emulate a 9436 EEPROM.

The ENC28J60 includes a generous set of configuration buttons and knobs. In addition to supporting Full and Half Duplex modes, the ENC28J60 enables you to select automatic retransmit on collision. There are also programmable On and Off switches for padding and CRC generation. Like the RTL8019AS and AX88796L, the ENC28J60 can deploy numerous filtering options under program control.

If you've ever dealt with the RTL8019AS, you know that it contains a 16-KB internal buffer area. In 8-bit mode, the application can use only 8 KB of the RTL8019AS's 16-KB

internal buffer. The ENC28J60 also incorporates 8 KB of internal buffer space.

Just like the RTL8019AS and AX88796L Ethernet engines, the internal transmit/receive buffer area is carved up into a circular receive buffer area and a linear transmit buffer area. The ENC28J60 hardware manages the flow of data from the 8-KB buffer area in much the same way as the RTL8019AS and AX88796L parts. An internal DMA/checksum engine with hardware buffer pointer auto-increment assistance streamlines the data movement processes. The ENC28J60 calls the buffer boundary identifiers and pointers by different names, but in reality, the ENC28J60 does it just like the RTL8019AS and AX88796L do it.

The ENC28J60's MAC subsystem can handle broadcast, unicast, and multicast packets. As I mentioned earlier, numerous receive packet filtering schemes can be deployed. In addition to the reception of packets with broadcast, unicast, and multicast addresses, the ENC28J60's MAC recognizes Magic Packet and hash table addressing. In power-critical applications, the host can be awakened when a receive filter packet true condition is met.

The RTL8019AS in particular doesn't enable easy access to its status LED options. Changing the ENC28J60's status LED behavior is a straightforward process in comparison. In fact, the way the ENC28J60's status LEDs are wired determines which duplex mode the ENC28J60 will initially operate in.

A 20-MHz clock drives the RTL8019AS. Like the AX88796L, the ENC28J60 requires a 25-MHz clock. The ENC28J60 is a 3.3-V part. Many of today's low-power microcontrollers can't be clocked as fast as their 5-V counterparts. That factor plays right into the hands of the ENC28J60's clock output pin, which provides a programmable clock source that could be used by a companion 3.3-V microcontroller or anything else that needs a clock.

I mentioned retrofitting mature microcontroller designs. If it's a 5-V design, the ENC28J60 can be easily incorporated because its inputs are all 5-V tolerant. The SPI data out pin (SO) and the two ENC28J60 interrupt outputs (INT and WOL) are not 5-V-compatible outputs and must be level shifted to work with 5-V systems. Fortunately, 3- to 5-V level shifting can be performed with simple circuitry based on standard HC-class logic.

If you want three-state capability, a 74HC125 will suffice as an output buffer. A simple 74HC08 interface between the ENC28J60's SO, INT, and WOL outputs and the microcontroller's 5-V I/O pins also would do the trick. We already know that the SPI consumes four of the ENC28J60's 28 pins. Refer to Figure 1 as I take you on a pin-by-pin tour of the remainder of the ENC28J60's I/O subsystem.

To avoid having to provide external 2.5-V power circuitry for some of the ENC28J60's internal Ethernet subsys-

tems, an on-chip 2.5-V regulator is realized by simply connecting a 10- μ F capacitor to pin 1 on the ENC28J60. All of the ENC28J60's V_{SS} pins (pins 2, 11, 18, 21, and 22) are tied to a common ground.

The state and output of the ENC28J60's CLKOUT pin are under the control of the bits in the ENC28J60's ECOCON register. With no clock division, the CLKOUT pin can provide a 25-MHz clock. Setting ECOCON bits to divide the clock by 2, 3, 4, or 8 provides a 12.5-, 8.333333-, 6.25-, or 3.125-MHz clock signal, respectively. The CLKOUT pin signal also can be disabled, which drives the CLKOUT pin to a logical low state.

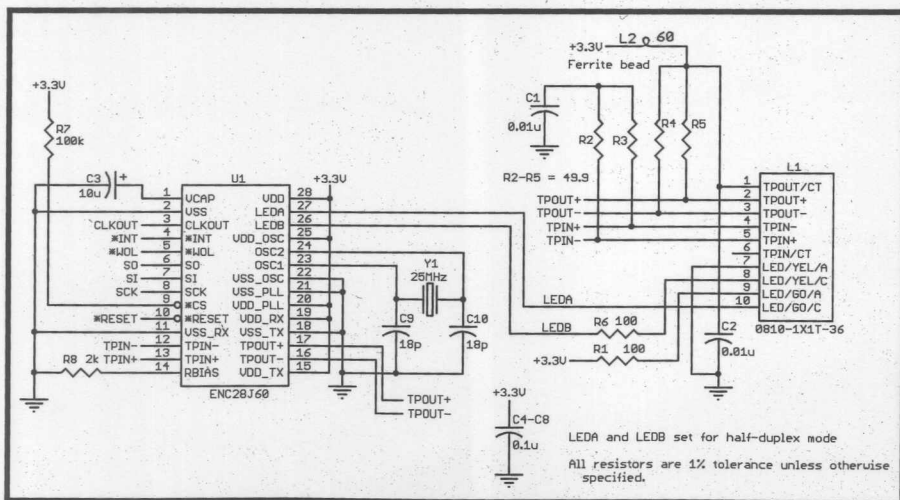
Multiple interrupt types stand behind the *INT pin. Basically, if the interrupt isn't related to wake-up on LAN (WOL) operations, it's presented via the ENC28J60's INT pin. The interrupt types that affect the INT pin include Receive Error, Transmit Error, Transmit Interrupt, Receive Packet Pending Interrupt, Link Change Interrupt, and DMA Interrupt. The interrupts are enabled, disabled, and recognized just as they are in microcontrollers with the assistance of interrupt flags and interrupt control registers. To service the interrupt flag and control structures more efficiently, the ENC28J60 incorporates special bit set (bit field set, or BFS) and bit clear (bit field clear, or BFC) instructions that can be issued via the SPI channel.

The WOL pin is also the front man for multiple WOL interrupt types. The WOL pin is primarily intended to awaken the host when one or a combination of the seven WOL interrupt types occurs. The WOL interrupt types include a wake-up on unicast packet, a wake-up on multicast packet, a wake-up on broadcast packet, a wake-up on any packet, a wake-up on magic packet, a wake-up on pattern match, and a wake-up on hash table.

The next four pins constitute the ENC28J60's SPI. Because the SPI clock isn't generated by the ENC28J60, it's considered to be a slave node. Note that the ENC28J60 pin names don't use the more meaningful master naming conventions. Pin 6, the SO pin, performs the master in slave out (MISO) function; it's the non-5-V-tolerant data out pin. The SI pin, which is a 5-V tolerant data input pin, performs the master out slave in (MISI) function. SPI data I/O clocking is synchronized on the ENC28J60's SCK pin, which is also a 5-V-tolerant input. Nothing happens with the SPI until the ENC28J60's chip select (*CS) is driven logically low. Note that the *CS pin is tied logically high to prevent any SPI pin funny business.

The ENC28J60's *RESET pin is identical in operation to a standard microcontroller reset input. You can either include a standard resistor capacitor/momentary switch circuit for the ENC28J60's *RESET pin or steal the reset signal from the existing microcontroller reset circuitry. I always use a blocking diode in my PIC reset circuits to prevent any high voltage from the MCLR pin from feeding back into the main power rail. That makes it easy to steal the reset logic levels by simply placing another blocking diode in parallel with the existing blocking diode and feeding the output of the added blocking diode to the ENC28J60 *RESET pin.

Pins 12 and 13 are differential receive pins. TPIN+ and TPIN- are connected to a 1:1 10BaseT pulse transformer. A center-tapped pulse transformer isn't required. However, the pair of 49.9- Ω precision resistors and the 0.01- μ F capacitor are required for proper termination.



The CS8900CQ Ethernet controller requires a 4.99-k Ω precision resistor to provide bias for the CS8900CQ's internal analog circuitry. Likewise, the ENC28J60 uses a 2-k Ω precision resistor attached to its RBIAS pin to activate the ENC28J60's internal analog subsystems.

TPOUT+ and TPOUT- are differential transmit pins that interface to a center-tapped 1:1 10BaseT-rated pulse transformer. Like the differential receive input pins (TPIN+ and TPIN-), a pair of 49.9- Ω precision resistors and a 0.1- μ F capacitor form the required termination circuitry. The center tap of the transmit pulse transformer is powered and a high-current 60- Ω ferrite bead is included to prevent noise from easily finding its way into the main 3.3-V power rail.

So far, the ENC28J60 I/O pins have taken on a microcontroller-like look and feel. That doesn't change with the ENC28J60's OSC1 and OSC2 pins. Figure 1 shows a typical parallel-cut crystal and dual-capacitor oscillator circuit. An external 3.3-V clock

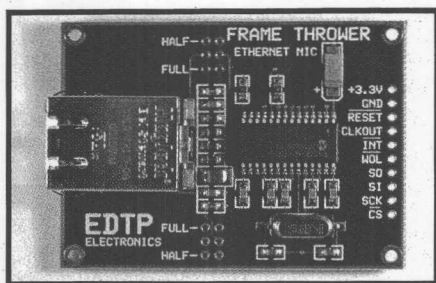


Photo 1—The SPI and low pin count make it easy to incorporate the ENC28J60 in any device that can support a SPI.

source can also drive OSC1.

LEDB can be configured via the PHLCON register to display transmit activity, receive activity, collision activity, link status, or duplex status. The LEDB I/O pin also can be configured to blink the attached LED in a fast or slow manner. In addition, the width of the LED flash can be stretched by 40, 73, or 139 ms by simply setting a couple of bits in the PHLCON register.

The LEDB connection is read at reset to determine how to initialize the PHCON1.PDPXMD (PHY Duplex mode) bit. If the LEDB pin sources

current to the attached LED, the PHY Duplex mode bit is cleared and Half Duplex mode is selected. Conversely, if the LEDB pin is sinking the attached LED current, the PHY duplex mode bit is set and Full Duplex mode is activated. In the event that no LED is attached, the PHY Duplex mode bit will be indeterminate at reset and you must issue indirect commands to the PHY register to initialize the bit. LEDA can be configured for the same operations as LEDB. However, LEDA doesn't figure into the duplex selection scheme.

That completes your tour of the ENC28J60 I/O. Before I move on to the next topic, I would like to point out that all of the ENC28J60's V_{DD} power pins are bypassed with a 0.1- μ F capacitor. With respect to the differential input and output pins, the perfect 10BaseT pulse transformer is the Bel Stewart MagJack 0810-1X1T-36, which also includes a pair of integral status LEDs.

GETTING PHYSICAL

So as it is drawn, so shall it be built. Behold Photo 1. The Frame Thrower is

POWER DISTRIBUTION and CONTROL from \$49.



PROTECT
YOUR
INVESTMENT



No Overload Protection
No Surge Protection
No Control Features
High Risk!

Over 40 Low Cost Models

- Circuit Breaker Protection
- Surge Protection
- Line Filtered
- 19" Rack Mount
- Bench Mount
- Order On-line



FREE SHIPPING
on most orders



www.bmpfpower.com Simplifying Power Distribution



a physical realization of the circuitry I described in Figure 1 during your tour of the ENC28J60 I/O. The ENC28J60 is available in a smaller 28-pin QFN package or a larger 28-pin DIP. I selected the 28-pin SOIC package because it's small enough to be considered compact and easy to handle and solder using standard hand-soldering tools. The Frame Thrower measures 2.5" x 1.8" and offers up the SPI, CLKOUT, *INT, *WOL, *RESET, and power connections on 10 0.1" center pads.

I constructed the Frame Thrower module shown in Photo 1 using single-sided SMT construction on a four-layer PCB. You can also build a Frame Thrower using conventional through-hole methods.

Note that I included jumper block pads to select either Full or Half Duplex mode. The jumper block pads are wired on the PCB to default to half-duplex operation by sourcing current to LEDB in the MagJack.

Another Ethernet device was born after the first Frame Thrower rolled

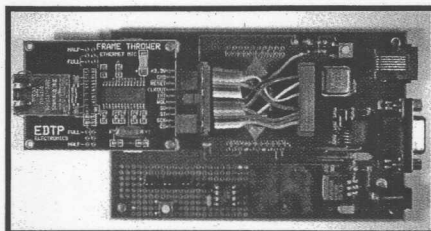


Photo 2—All it takes are 10 connections to wire the Frame Thrower into just about any device that supports a SPI. The Microchip TCP/IP stack makes things easy on the ENC28J60 firmware/driver side. Note the extra RESET-stealing diode just above the IDC connector.

out of the reflow oven. However, the Frame Thrower by itself in this form was just a pretty, little, useless piece of electronic gear. The Frame Thrower needed a suitable host. To keep my construction workload at a minimum, I decided to select and retrofit one of the many prototype boards in the Florida room. I ruled out any 5-V prototypes because I didn't want to add a level shifter IC at the time. Ideally, I wanted to be able to simply plug the Frame Thrower into a 3.3-V host. As it turned out, the AirDrop-P was the perfect Frame Thrower host. As you

can see in Photo 2, the AirDrop-P prototype PCB contains a regulated 3.3-V power supply, a serial port, a reset circuit, a programming/debugging port, a SPI-capable microcontroller, and some space for additional goodies.

The AirDrop-P prototype PCB in Photo 2 is based on a PIC18LF8621 microcontroller, which contains integral SPI hardware. All of the PIC18LF8621's I/O pins are terminated at 0.1" header points. I soldered in a dual row of 10 standard 0.1" center pins in the PIC18LF8621's SPI I/O pin area and added a standard 0.1" center 20-pin male connector to the Frame Thrower's I/O interface pads. I then simply cabled in the SPI between the PIC18LF8621 and the Frame Thrower in the normal manner (ENC28J60 SO to PIC SDI, ENC28J60 SI to PIC SDO, and ENC28J60 SCK to PIC SCK) using a 20-pin IDC connector at both ends of the ribbon cable.

I connected the ENC28J60's *INT and *WOL pins to the PIC18LF8621's RB0 and RB1 external interrupt input pins. The ENC28J60's SPI CS line was

PCB-POOL®

Instant online Quotes
Tooling and set-up included!
No Minimum Quantity,
No drill Limitations!
Full DRC on all orders +
Top Quality and on time
delivery Guaranteed!
 (if its late, get it free!)

Price Example: 16 Sq-Inches, (ds - pth)

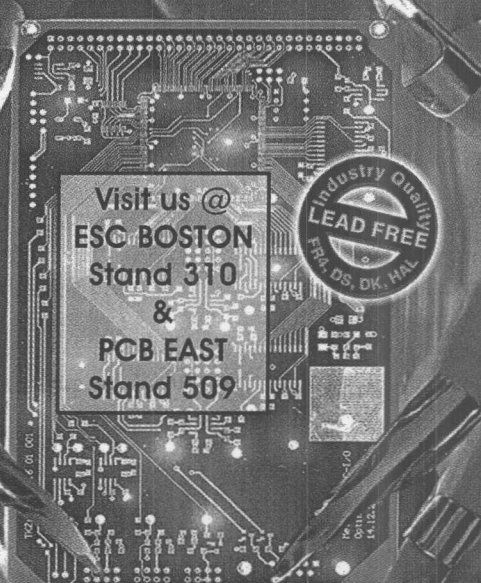
2 Days: \$ 90

8 Days: \$ 22.5

Standard PCBPool Service
 SIMPLY SEND YOUR FILES AND ORDER ONLINE!

WWW.PCBPOOL.COM

DOWNLOAD OUR FREE PCB SOFTWARE
www.free-pcb-software.com



TOLLFREE: 1877 3908541
sales@beta-layout.com

tied to the PIC18LF8621's RB3 I/O pin. Of course, I tapped the Frame Thrower power pins into the AirDrop-P's 3.3-V power grid. If you look closely at Photo 2 just above the IDC connector on the AirDrop-P prototype PCB, you'll see that I added a second blocking diode in parallel with the existing blocking diode to steal the microcontroller's reset signal for the Frame Thrower's *RESET pin. As luck would have it, the Frame Thrower's mounting holes matched up exactly to the CompactFlash connector mounting holes on the AirDrop-P prototype PCB. That's it. The Frame Thrower was retrofitted on a mature AirDrop-P prototype.

WHAT FIRMWARE?

You've probably been waiting anxiously for me to address the firmware. You want to see the firmware that drives the Frame Thrower, right? Sorry. I didn't write any. The Frame Thrower uses the firmware contained in the free Microchip TCP/IP stack. Microchip even supplied the ENC28J60 driver module.

I studied the ENC28J60 driver and the Microchip stack and determined that I needed to make a couple of small changes in the original stack code. I also had to add some LEDs, a push button switch, and a 32-KB SPI EEPROM (25LC256) to take advantage of the stack's built-in goodies.

The original stack code assumes a 10-MHz clock driving a 40-pin PIC18F4620. My AirDrop-P prototype contains an 80-pin PIC18LF8621 running at 20 MHz. Because the Microchip stack calculates timings and data rates from a clock speed variable (CLOCK_FREQ), I simply made the clock speed change in the proper stack module (Compiler.h), as shown in Listing 1.

I'm extremely fond of Hi-Tech Software's PICC-18 C compiler, and I consider myself fortunate that the Microchip stack was written to support the PICC-18 C compiler as well as Microchip's C18. As you can see in the middle portion of Listing 1, that made adding my fuse settings for the

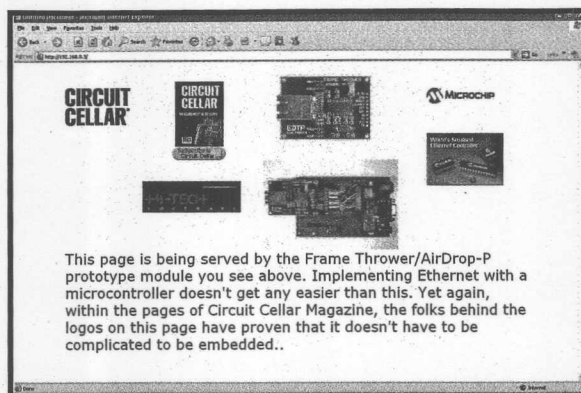


Photo 3—This page says it all. The process was painless from beginning to end.

PIC18LF8621 in the MainDemo.c module child's play. Coding the Frame Thrower was that simple.

THROWING FRAMES

I'm loving this. After I had the hardware together and the firmware was ready to roll with minor coding changes, it was time to power everything up, compile and load the stack, and throw some Ethernet frames around the Florida room.

The MainDemo.c module wants to see an LED attached to I/O pins RA2, RA3, and RA4. Also, a normally open momentary push button switch should be attached to I/O pin RB5 to enable entry into a configuration mode supported by the RS-232 interface. I could have hooked up a couple of 10-k Ω potentiometers to RA0 and RA1, but I didn't do that in this spin. The potentiometers act as voltage dividers to feed the PIC18LF8621 microcontroller's ADC. The LED attached to RA4 should blink in 1-s intervals when the stack is operating correctly. I also added the 25LC256 EEPROM to store web pages for the stack's HTTP functionality.

I attached an MPLAB ICD 2 to the AirDrop-P/Frame Thrower module, connected the module's serial port to a PC's COM1, tied the Frame Thrower to an Ethernet switch port with some twisted pair cable, and applied power to the entire thing. As I had expected, the Frame Thrower appeared to be dead and there wasn't an active status LED to be found. The modified stack compiled with no problems and the MPLAB ICD 2 loaded the resultant hex file into the AirDrop-P's on-board PIC18LF8621. I opened a HyperTerminal

window and told the MPLAB to turn her loose.

One of the three LEDs I added to the AirDrop-P prototype PCB began to blink in 1-s intervals. Shortly thereafter, I was informed via the HyperTerminal window that the Frame Thrower had requested an IP address via DHCP and was given 192.168.0.3. I pinged 192.168.0.3 and received positive results. The Frame Thrower was online!


The IP address lease and the successful ping meant ARP, DHCP, and ICMP were all working. I could read the information in the HyperTerminal window, which meant the RS-232 port was initialized and functioning properly.

The next logical thing to do was to attempt to load a canned web page into the EEPROM using FTP. I chose a web page that enabled the manipulation of the LEDs attached to I/O pins RA2 and RA3. The page also monitors the analog-to-digital voltage inputs and the state of the push button switch.

The EEPROM FTP download went off without a hitch and I found myself toggling the LEDs on the AirDrop-P prototype module with mouse clicks. But the stack kept crashing intermittently. That wasn't good. After consulting with an ENC28J60 guru at Microchip, I made one more minor modification to the Microchip stack that ultimately solved the crashing problem. I had been pushing the SPI too fast. The fix was to change the SPI Master mode clock speed to $F_{osc}/16$ from the original $F_{osc}/4$. The revised SSPCON1 value for $F_{osc}/16$ is at the bottom of Listing 1.

FRAMING OUT

I ran the Frame Thrower hardware and the modified stack continuously for a week. The crashing had stopped and everything was working as advertised. This was by far the easiest Ethernet implementation I'd ever done from scratch. You should be able to find all of the datasheet data and firmware on Microchip's web site, but if you have any difficulties, just drop me a line and I'll help you out.

A neat little web page utility was included in my stack package that takes everything in a directory and converts it to a binary image that can be downloaded via FTP or the RS-232 port to the EEPROM as a web page. I'll leave you with the thought conveyed in Photo 3. 

Fred Eady has more than 20 years of experience as a systems engineer. He has worked with computers and communication systems large and small, simple and complex. His forte is embedded-systems design and communications. Fred

may be reached at fred@edtp.com.

SOURCES

AirDrop-P/Frame Thrower module
EDTP Electronics
www.edtp.com

PICC-18 C COMPILER
HI-TECH Software
www.htsoft.com

ENC28J60 Ethernet controller and PIC18LF8621 microcontroller
Microchip Technology
www.microchip.com

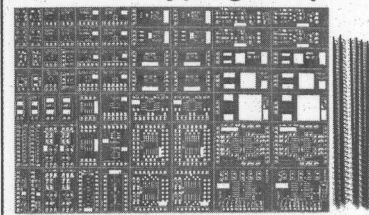
Listing 1—All I had to do was find the variables I needed to change and recompile. The Microchip stack is modular and easy to navigate.

```
//In compiler.h
//Clock frequency value.
//This value is used to calculate Tick Counter value
#define CLOCK_FREQ (20000000) //Hertz
//IN MainDemo.c **
//Set configuration fuses for a HI-TECH compiler. For an MCC18 compiler,
//separately linked C18cfg.asm file will set the correct fuses.
#if defined(HITECH_C18)
//Debug fuses
__CONFIG(1, OSCDIS & HS);
__CONFIG(2, BORDIS & PWRTDIS & WDTDIS);
__CONFIG(3, MCU & MCLREN);
__CONFIG(4, DEBUGEN & LVPDIS & STVREN);
__CONFIG(5, UNPROTECT);
__CONFIG(6, WRNEN);
//Settings for PIC18F4620
//__CONFIG(1, UNPROTECT & 0x32FF); //Use Fail-safe clock monitor disable,
//oscillator switch over disabled, HS
//__CONFIG(1, UNPROTECT & 0x36FF); //Failsafe clock monitor disable,
//oscillator switch over disabled, HS_PLL
//__CONFIG(1, UNPROTECT & 0x35FF); //Failsafe clock monitor disable,
//oscillator switch over disabled, ECRA6
//__CONFIG(2, PWRTDIS & BORDIS & WDTDIS); //Use
// Old settings for PIC18F452
//__CONFIG(1, UNPROTECT & HS);
//__CONFIG(2, PWRTEN & BORDIS & WDTDIS);
#endif
//In MACInit
void MACInit(void)
{
    BYTE i;
    //Set up the SPI module on the PIC for communications with the ENC28J60
    SPIUnselectEthernet();
    MCP_CS_TRIS = 0; //Make the Chip Select pin an output
    TRISC_RC3 = 0; //Set RC3 (SCK) pin as an output
    TRISC_RC4 = 1; //Make sure RC4 (SDI) pin is an input
    TRISC_RC5 = 0; //Set RC5 (SDO) pin as an output
    SSPCON1 = 0x21; //SSPEN bit is set, SPI in master mode,
    //FOSC/4, IDLE state is low level
    //changed to FOSC/16 with 20-MHz clock

    PIR1_SSPIF = 0;
    SSPSTAT_CKE = 1; //Transmit data on rising edge of clock
    SSPSTAT_SMP = 0; //Input sampled at middle of data
    //output time

    //Wait for CLKRDY to become set. Bit 3 in ESTAT is an unimplemented bit. If
    //it reads out as "1," that the part is in RESET or otherwise the SPI pin
    //is being driven incorrectly. Make sure it is working before proceeding.
    do
    {
        i = ReadETHReg(ESTAT).Val;
    } while((i & 0x08) || (~i & ESTAT_CLKRDY));
```

SMT Prototyping Adapters



Snap-Apart™ PCB's with .100" pin strips
Dozens of assorted adapters on each PCB
A variety of PCB's - patterns on both sides
Eliminate the cost and delay of PCB layout / fab!

SOIC PLCC SSOP QFP DPAK
TSSOP SOT23 MSOP QSOP
SC90 D2PAK SOT89 SOT143
TSOP SC70 SC88 SC326
and many more.



BELLIN DYNAMIC SYSTEMS, INC.

www.beldynsys.com (714) 630-8024

- Rapid Development Solutions for the Technical Professional -

PROFESSORS

The Circuit Cellar college program
puts quality engineering information
in the hands of your students every
month. Sign up now to get
Circuit Cellar distributed to your
class this semester.



To update your professor account or to find
out more about our college program, visit
www.circuitcellar.com/products/collegeprogram

PIC-SERVO MOTION CONTROL

MOTION CONTROLLERS FOR
BRUSH, BRUSHLESS AND
STEPPER MOTORS.

- CONTROLLER CHIPS
- CONTROLLER BOARDS

www.picservo.com

JEFFREY KERR, LLC